Universitat Pompeu Fabra Barcelona

Lecture 14 · Communities

Networks, Crowds and Markets

What is coming next

1. What do we mean by a community?
2. Zachary's Karate Club
3. Hypotheses and definitions (H1–H4)
4. The Girvan–Newman algorithm
5. The Stochastic Block Model (SBM)

Communities in graphs

What are communities?

Definition (Informal)

Groups of nodes with more connections inside than outside.

Examples:

- Social networks: circles of friends, political communities.
- Scientific collaboration: fields or subdisciplines.
- Biology: protein complexes in interaction networks.
- Infrastructure: airline networks with hubs and regional groups.

There is no single formal definition.

Communities in Economics

Trade Blocs: Countries cluster into EU, NAFTA, ASEAN.

Political Polarization: Twitter users cluster into left vs. right.

Firms: Industry supply chains reveal modular communities.

- Detecting communities = identifying hidden structure in markets.

A famous example: Communication structure in Belgium.

Zachary's Karate Club

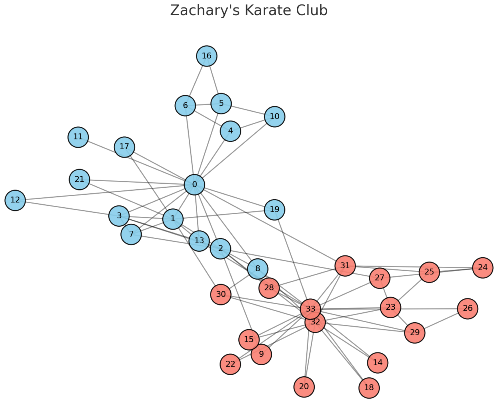
This is the first famous example of community structure in a network.

Initially analysed by Wayne W. Zachary, 1977.

Became the most classic network for community analysis.

- 34 members of a karate club, edges = friendships outside the club.
- Conflict between instructor ("Mr. Hi") and administrator ("John A") led to a split of the club.
- Based on the friendship network, predict how the club splits.
- Zachary's analysis correctly predicted all but one member's side.

Karate Club Network

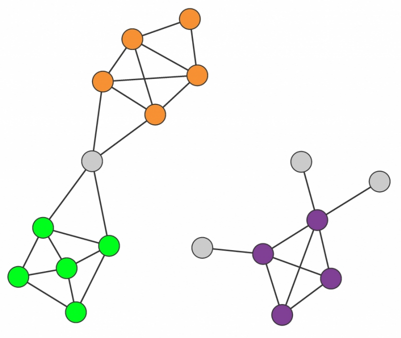


- Mr. Hi corresponds to node 0,
- John A corresponds to node 33.

Communities: Defining principles

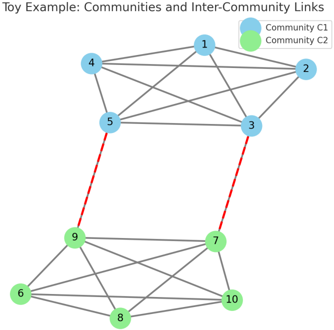
Principle 1 – Fundamental

A network's community structure is encoded in its wiring diagram. That is, communities can in principle be discovered by looking only at the graph structure.



Principle 2 – Connectedness & Density

A **community** should be a connected subgraph. Links inside a community should be denser than links going outside.



Strong vs Weak Communities

$$\deg_C(v) = \#\{\text{edges from } v \text{ to nodes in } C\}$$

$$\deg_{\overline{C}}(v) = \#\{\text{edges from } v \text{ to nodes outside } C\}.$$

Strong vs Weak Communities

$$\deg_C(v) = \#\{\text{edges from } v \text{ to nodes in } C\}$$

$$\deg_{\overline{C}}(v) = \#\{\text{edges from } v \text{ to nodes outside } C\}.$$

Strong community (restrictive): every node $v \in C$ satisfies

$$\deg_C(v) \; > \; \deg_{\overline{C}}(v).$$

Strong vs Weak Communities

$$\deg_C(v) = \#\{\text{edges from } v \text{ to nodes in } C\}$$
$$\deg_{\overline{C}}(v) = \#\{\text{edges from } v \text{ to nodes outside } C\}.$$

Strong community (restrictive): every node $v \in C$ satisfies

$$\deg_C(v) \; > \; \deg_{\overline{C}}(v).$$

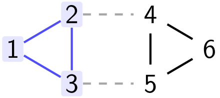
Weak community: total internal degree exceeds external degree:

$$\sum_{v \in C} \deg_C(v) \; > \; \sum_{v \in C} \deg_{\overline{C}}(v).$$

(the average in-community degree larger than out-community degree)

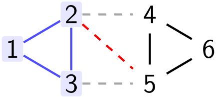
Example · Strong vs Weak Community

Original graph (strong)



$C = \{1, 2, 3\}$ $\overline{C} = \{4, 5, 6\}$

After adding edge $(2, 5)$



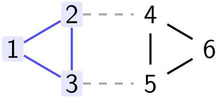
$C = \{1, 2, 3\}$ $\overline{C} = \{4, 5, 6\}$

v	$\deg_C(v)$	$\deg_{\overline{C}}(v)$	Strong?
1	2	0	✓
2	2	1	✓
3	2	1	✓

$\Rightarrow$ C is a strong community.

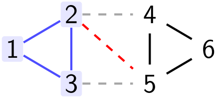
Example · Strong vs Weak Community

Original graph (strong)



$C = \{1,2,3\}$ $\overline{C} = \{4,5,6\}$

After adding edge $(2,5)$



$C = \{1,2,3\}$ $\overline{C} = \{4,5,6\}$

v	$\deg_C(v)$	$\deg_{\overline{C}}(v)$	Strong?
1	2	0	✓
2	2	1	✓
3	2	1	✓

$\Rightarrow$ C is a strong community.

Now add edge $(2,5)$: $\deg_C(2) = 2$, $\deg_{\overline{C}}(2) = 2 \Rightarrow$ not strong. But

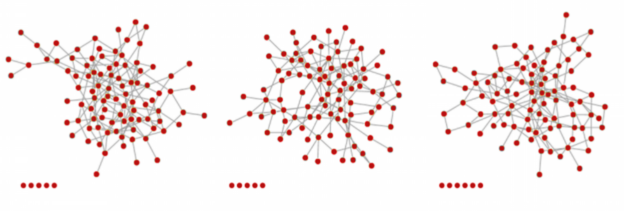
$$\sum_{v \in C} \deg_C(v) = 6, \qquad \sum_{v \in C} \deg_{\overline{C}}(v) = 4,$$

so C is still a weak community.

Principle 3 – Random Baseline

Erdős–Rényi graphs do not have meaningful community structure.

- Communities are detected when the observed structure **deviates significantly from total randomness**.
- Useful for benchmarking algorithms for community detection.



Modularity

Modularity Maximization

Connections *within communities* denser than expected by random chance.

Measured by modularity (Newman-Girvan 2004):

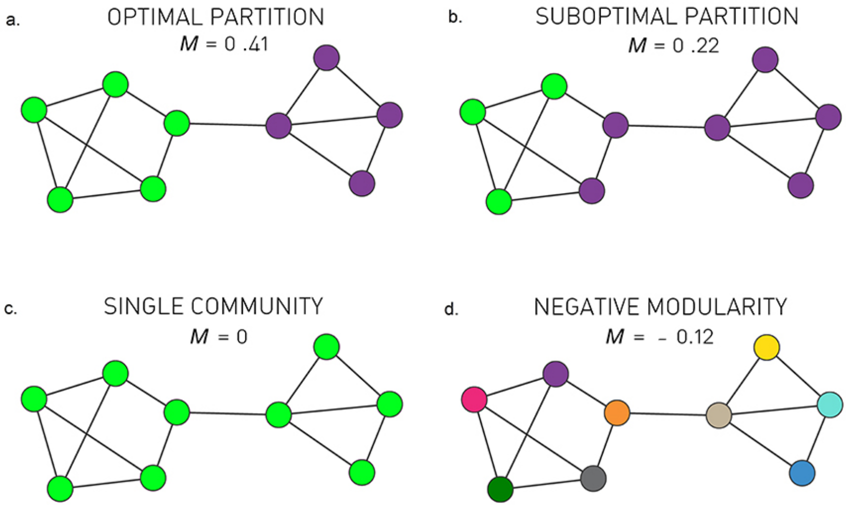
$$M \;=\; \frac{1}{2L}\sum_{i,j}\Big(A_{ij}-\frac{k_i k_j}{2L}\Big)\delta(c_i,c_j),$$

where

- A_{ij} - adjacency matrix (1 if edge i–j exists, else 0),
- k_i - degree of node i, $2L=\sum_i k_i$ (total edge ends),
- $\delta(c_i,c_j)=1$ if i,j lie in the same community, 0 otherwise.

M compares the *observed* edges (A_{ij}) to what we would *expect at random* ($k_i k_j/2L$; see configuration model).

The Girvan-Newman Algorithm



Identifying communities

Community detection via hierarchical clustering

Find a computationally efficient community detection procedure.

Let $S = [\delta_{ij}]$ be a similarity matrix:

- S symmetric;
- $s_{ij} \geq 0$ for all $i \neq j$.

If i, j are "close", s_{ij} is higher.

- Build a *similarity matrix* s_{ij} from the network.
- Iteratively group (or split) nodes using s_{ij}.
- Output is a dendrogram; cutting it gives a partition.

An agglomerative algorithm

Ravasz agglomerative algorithm

Let $B_i := \{j : d(i,j) \leq 1\}$. Let A be the adjacency matrix.

We define node similarity using the *topological overlap*:

$$s_{ij} \;=\; \frac{|B_i \cap B_j|}{\min\{|B_i|, |B_j|\}} \;\in\; [0,1], \qquad (i \neq j).$$

- $s_{ij} = 0$ iff i, j are not connected and they share no neighbors.
- $s_{ij} = 1$ iff $B_i \subseteq B_j$ or $B_j \subseteq B_j$.

Ravasz agglomerative algorithm

Let $B_i := \{j : d(i,j) \leq 1\}$. Let A be the adjacency matrix.

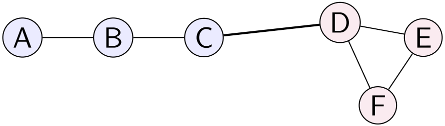
We define node similarity using the *topological overlap*:

$$s_{ij} \ = \ \frac{|B_i \cap B_j|}{\min\{|B_i|, |B_j|\}} \ \in \ [0,1], \qquad (i \neq j).$$

- $s_{ij} = 0$ iff i, j are not connected and they share no neighbors.
- $s_{ij} = 1$ iff $B_i \subseteq B_j$ or $B_j \subseteq B_j$.

Many authors subtract A_{ij} in both the nominator and denominator to slightly down-weight the direct edge when i and j are linked ($\tilde{s}_{ij}$).

Toy example



Pair	$\lvert B_i \rvert$	$\lvert B_j \rvert$	$\lvert B_i \cap B_j \rvert$	s_{ij}	$\tilde{s}_{ij}$	A_{ij}
A–B	2	3	2	1.00	1.00	1
B–C	3	3	2	0.67	0.50	1
C–D (bridge)	3	4	2	0.67	0.33	1
D–E	4	3	3	1.00	1.00	1
D–F	4	3	3	1.00	1.00	1
E–F	3	3	3	1.00	1.00	1
A–C	2	3	1	0.50	0.50	0
B–D	3	4	1	0.33	0.33	0
C–E	3	3	1	0.33	0.33	0
A–D	2	4	0	0.00	0.00	0
A–E	2	3	0	0.00	0.00	0
A–F	2	3	0	0.00	0.00	0

Ravasz agglomerative algorithm: hierarchical clustering

We apply standard hierarchical clustering to S.

Algorithm.

1. Compute the similarity matrix $s_{ij} = \frac{|B_i \cap B_j|}{\min(|B_i|, |B_j|)}$ for i, j.
2. Treat each node as a separate cluster.

Ravasz agglomerative algorithm: hierarchical clustering

We apply standard hierarchical clustering to S.

Algorithm.

1. Compute the similarity matrix $s_{ij} = \frac{|B_i \cap B_j|}{\min(|B_i|,|B_j|)}$ for i, j.
2. Treat each node as a separate cluster.
3. Find the two clusters with the largest average pairwise s_{ij}.
4. Merge them into a new cluster. Compute similarities between this new cluster and every other cluster using a chosen linkage rule:

$$s_{A,B} = \begin{cases} \max_{i \in A, j \in B} s_{ij}, & \text{(single linkage)}, \\ \min_{i \in A, j \in B} s_{ij}, & \text{(complete linkage)}, \\ \text{average}_{i \in A, j \in B} s_{ij}, & \text{(average linkage)}. \end{cases}$$

Ravasz agglomerative algorithm: hierarchical clustering

We apply standard hierarchical clustering to S.

Algorithm.

1. Compute the similarity matrix $s_{ij} = \frac{|B_i \cap B_j|}{\min(|B_i|,|B_j|)}$ for i, j.
2. Treat each node as a separate cluster.
3. Find the two clusters with the largest average pairwise s_{ij}.
4. Merge them into a new cluster. Compute similarities between this new cluster and every other cluster using a chosen linkage rule:

$$s_{A,B} = \begin{cases} \max_{i \in A, j \in B} s_{ij}, & \text{(single linkage)}, \\ \min_{i \in A, j \in B} s_{ij}, & \text{(complete linkage)}, \\ \text{average}_{i \in A, j \in B} s_{ij}, & \text{(average linkage)}. \end{cases}$$

5. Repeat Step 3–4 until all nodes are merged into one cluster.

Output. The sequence of merges defines a *dendrogram*. Cutting it at a chosen similarity threshold yields the community partition.

Next week we will discuss in detail an example.